

The Bioinformatics Lab

Linux proficiency
terminal-based text editors
version control systems

Jonas Reeb

30.04.2013

“What makes you proficient on the command line?” - General ideas

- ▶ Use CLIs in the first place
- ▶ Use each tool for what it does best
- ▶ Chain tools for more complex tasks
- ▶ Use power of shell for small scripting jobs
- ▶ Automate repeating tasks
- ▶ Knowledge of regular expression

Standard tools

- ▶ man
- ▶ ls/cd/mkdir/rm/touch/cp/mv/chmod/cat...
- ▶ grep, sort, uniq
- ▶ find
- ▶ wget/curl
- ▶ scp/ssh
- ▶ top(/htop/iftop/iotop)
- ▶ bg/fg

Input-Output Redirection I

By default three streams (“files”) open

Name	Descriptor
stdin	0
stdout	1
stderr	2

Any program can check for its file descriptors' redirection! (`isatty`)

Input-Output Redirection II

Output

- ▶ `M>f` Redirect file descriptor `M` to file `f`, e.g. `1>f`
- ▶ Use `>>` for appending
- ▶ `&>f` Redirect `stdout` and `stderr` to `f`
- ▶ `M>&N` Redirect fd `M` to fd `N`

Input

- ▶ `0<f` Read from file `f`

Pipes

- ▶ Forward output of one program to input of another
- ▶ Essential for Unix philosophy of specialized tools
- ▶ `grep -P -v "^>" *.fa | sort -u > seqs`
- ▶ Input and arguments are different things. Use `xargs` for arguments: `ls *.fa | xargs rm`

Scripting

- ▶ Quick way to get basic programs running
- ▶ Basic layout:

```
#!/bin/bash
if test "$1"
then
    count=$1
else
    count=0
fi
for i in {1..10}
do
    echo $((i+count))
    let "count += 1"
done
```












Motivation - “What makes a good text editor”

- ▶ Fast execution, little system load
- ▶ Little bandwidth needed
- ▶ Available for all (your) major platforms
→ Familiar environment
- ▶ Fully controllable via keyboard
- ▶ Extensible and customizable
- ▶ Auto-indent, Auto-complete, Syntax highlighting, Folding, ...

Vim - Modal editing




- ▶ Vim is modal, keystrokes have differing effects depending on which mode you are in
- ▶ Normal mode: For thinking, reading, navigating, deleting, copying, pasting, moving,
- ▶ Insert mode: Probably Vim's least powerful mode. Essentially as good as any other editor

Vim - Movement

- ▶ Move small distances with  ,  ,  , 
- ▶ Move to character within line, using  , 
- ▶ Move word-wise using  ,  ( , )
- ▶ Use  with a number to jump to specific lines
- ▶ To move screen lines instead of file lines:




```
noremap j gj  
noremap k gk  
noremap j gj  
noremap k gk
```

Vim - Searching

- ▶ Use search via  for larger motions
- ▶ Jump through matches with , 
- ▶ Matches stay intact, even after entering insert mode
- ▶ Some useful settings:

```
"Everything other than a-zA-Z0-9_ is treated as special character
noremap / \v
set hlsearch      " highlight  searches
set incsearch     " incremental search while typing
set ignorecase    " Do case insensitive  search
set smartcase     " Don't ignore case if at least one letter upper case
set gdefault      " always do global search and replace
```

Vim - (Persistent) undo

- ▶ Use  to undo,  -  to redo
- ▶ Since Vim 7.3 the undo stack can be written to a file by setting:

```
set undofile
```

- ▶ Will result in:

```
-rw-rw-r-- 1 jonas jonas 33 Apr 29 16:25 script.py  
-rw-rw-r-- 1 jonas jonas 1389 Apr 29 16:25 .script.py.un~
```

Motivation - “What is the purpose of version control systems?”

- ▶ Backups
- ▶ Retraceability, Rollbacks
- ▶ Collaboration, Synchronization
- ▶ Parallelism by branching
- ▶ Keeping multiple versions
- ▶ Orderly (, well documented) changes

VCS Commands - Status

- ▶ Show status of tracked and untracked files
 - Summarizes what next commit will record

```
→ presentation git:(master) X git status -s
A  added
AM added_and_changed
M  changed
D  deleted
?? untracked
```

VCS Commands - Add

- ▶ Add a file/folder to version control
- ▶ Some might be automatically ignored

```
→ presentation git:(master) ✗ touch doc1 doc2 doc1~
```

```
→ presentation git:(master) ✗ git add doc*
```

```
The following paths are ignored by one of your .gitignore files:  
doc1~
```

```
Use -f if you really want to add them.
```

VCS Commands - Remove

- ▶ Remove something from the VCS' tracking
- ▶ Depending on VCS might also remove the file from your local disk
- ▶ Deleting file on disk will likely have the same effect

VCS Commands - Commit

- ▶ Record your changes to the repository
- ▶ A file state you can revert to later on
- ▶ Logically group files and supply useful commit messages

```
→ presentation git:(master) ✗ git commit -a -m 'change sth'  
[master 4a5f5a9] change sth  
1 file changed, 3 insertions(+), 1 deletion(-)
```

VCS Commands - Log

- Show a timeline of commits

```
* d7d7271 (HEAD, origin/master, origin/HEAD, master) Merge branch 'v4l_for_linus' of git://git.k
-media Linux Torvalds, 3 days ago
| \
| * c95789e [media] cx25821: do not expose broken video output streams Hans Verkuil, 2 weeks ago
| * f1b0c82 [media] mb86a20s: Fix estimate_rate setting Mauro Carvalho Chehab, 4 weeks ago
| * 96edcf3 Merge branch 'fixes-3.9-late' of git://git.kernel.org/pub/scm/linux/kernel/git/delle
| \ \
| * bda079d parisc: use spin_lock_irqsave/spin_unlock_irqrestore for PTE updates John David Ang
| * cf71130 parisc: disable -mlong-calls compiler option for kernel modules Helge Deller, 6 day
| * 0f28b62 parisc: uaccess: fix compiler warnings caused by __put_user casting Will Deacon, 7
| * 87be2f8 parisc: Change kunmap macro to static inline function John David Anglin, 6 days ago
| * ca0ad83 parisc: Provide __ucmpdi2 to resolve undefined references in 32 bit builds. John Da
| * f464246 efivars: only check for duplicates on the registered list Matt Fleming, 3 days ago
| * 37b7f3c TTY: fix atime/mtime regression Jiri Slaby, 3 days ago
| * 91d80a8 aio: fix possible invalid memory access when DEBUG is enabled Zhao Hongjiang, 3 day
| * 697dfd8 Merge tag 'efi-urgent' into x86/urgent H. Peter Anvin, 4 days ago
| \ \ \
| | / /
| / |
| * f697036 efi: Check EFI revision in setup_efi_vars Josh Boyer, 5 days ago
```

VCS Commands - Diff

- Show line by line details of status command

```
diff --git a/terminator/config b/terminator/config
index d719305..df2bfbf 100644
--- a/terminator/config
+++ b/terminator/config
@@ -1,8 +1,10 @@
+[global_config]
+[keybindings]
 [profiles]
- [[default]]
-   font = Envy Code R 10
-   background_color = "#000000" # A comment
-   foreground_color = "#FFFFFF" # Note that hex colour values must be quoted
+ [[default]]
+   use_system_font = False
+   font = Envy Code R 22
+   foreground_color = "#ffffff"
```

VCS Commands - Revert

- ▶ SVN: Undo local changes
- ▶ Git:
 - ▶ Revert: Undo a commit
 - ▶ Reset: Return to a previous state
 - ▶ Checkout: Undo changes in single files (also switches between branches)

VCS Commands - Merge

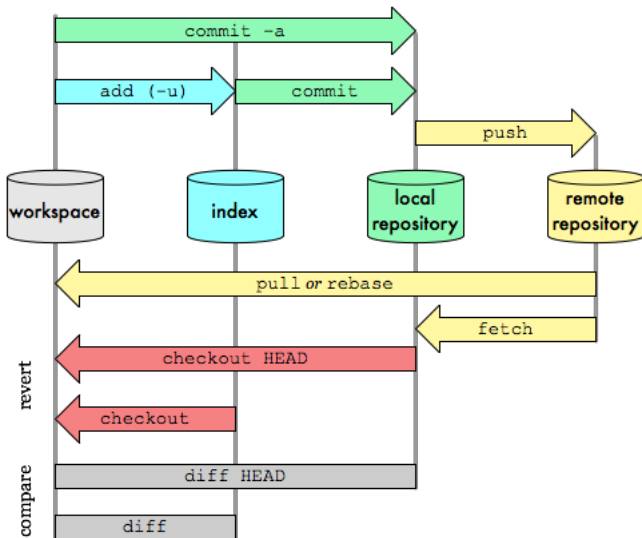
- ▶ Combining code from different branches or versions
- ▶ VCS will try to automate it
- ▶ Conflicts are recorded in the respective files:

```
This is an unaffected line
<<<<<<< HEAD
This line was changed on the master branch
=====
This is the same line, changed on the branch 'test'
>>>>>>> test
An unaffected line
```

Git

Git Data Transport Commands

<http://ostelee.com>



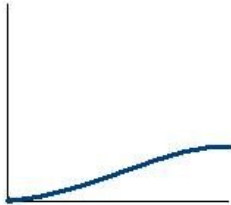
Thank you

Classical learning
curves for some
common editors

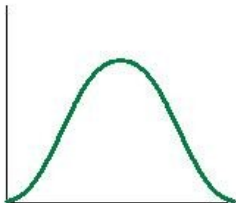
Notepad



Pico



Visual Studio



vi



emacs



Resources - Command line - References only

- ▶ <http://www.tldp.org/LDP/GNU-Linux-Tools-Summary/html/GNU-Linux-Tools-Summary.html>
- ▶ http://en.wikipedia.org/wiki/Unix_philosophy
- ▶ <http://tldp.org/LDP/abs/html/>

Resources - Vim - References only

- ▶ <http://pragprog.com/book/dnvim/practical-vim> (I own this, email me for the .pdf whoever wants to take a look at it)
- ▶ <http://vimcasts.org/>

Resources - Git - References only

- ▶ <http://rogerdudler.github.io/git-guide/>
- ▶ <http://gitref.org/basic/>
- ▶ <http://gitready.com/>

Resources - Command line I

- ▶ Common Linux command line tools
- ▶ Unix philosophy
- ▶ A good book about regular expressions
- ▶ Curl vs. Wget
- ▶ XKCD on tar

Resources - Command line II

- ▶ Bash I/O Redirection
- ▶ For toying around with redirection (-t calls isatty internally):

```
1 print STDOUT 'Here is some stdout';
2 if(! -t STDOUT) { print STDOUT " for redirection\n"; } else { print
  STDOUT "\n"; }
3 print STDERR 'Here is some stderr';
4 if(! -t STDERR) { print STDERR " for redirection\n"; } else { print
  STDERR "\n"; }
5 while(<STDIN>) { print; } #End with <C-d>
```

- ▶ Bash scripting

Resources - Vim I

- ▶ My dotfiles including `.vimrc`
- ▶ Video: “Dick size war for nerds” Vim vs. Emacs vs. Sed
- ▶ A good (e)Book on Vim
- ▶ Screencasts on Vim
- ▶ Vim cheat sheet
- ▶ VimGolf

Resources - Git I

- ▶ Short Git introduction
- ▶ Slightly longer Git introduction
- ▶ Another Git learning resource
- ▶ How to format Git commit messages
- ▶ Git GUI for Linux
- ▶ Git plugin for Vim